COUNTERPLANNING FOR MULTI-AGENT PLANS USING STOCHASTIC MEANS-ENDS ANALYSIS

Neil C. Rowe
Modeling, Simulation, and Virtual Environments Institute, U.S. Naval Postgraduate School
Code CS/Rp, 833 Dyer Road, Monterey CA 93943
USA

Sylvio F. Andrade
Brazilian Navy
Av. Canal de Marapendi 1700/1306, Barra da Tijuca, Rio de Janeiro, RJ, CEP 22631-050
Brazil

Abstract

We have developed a hierarchical planning method for multiple agents in worlds with significant levels of uncertainty. This has resulted in expert-system tools (MEAGENT) for analysts and planners without background in artificial intelligence. MEAGENT is particularly useful in analysis of counterplanning methods intended to thwart plans in complex situations. We apply heuristics to define experiments involving many runs of carefully modified simulations, use the results to quantify the effects of various counterplanning tactics, and then produce a counterplan. We exemplify our "experimental AI" approach for the domain of firefighting on ships.

Key Words: Counterplanning, planning, intelligent agents, forecasting and prediction, firefighting

1. The Planner

Our multi-agent planner for stochastic situations MEAGENT is a Prolog system using an old idea in artificial intelligence, means-ends analysis [8] as well as more modern ideas [5]. Means-ends is a form of goal-directed behavior that hierarchically decomposes plans into precondition and postcondition subtrees. This requires formal definition of actions or "operators" by their preconditions, deletion and addition postconditions, and most importantly, recommendation conditions [12]. A plan is represented as a tree and decomposed until null subtrees, or tasks involving only preconditions already achieved, are present at all leaves. Means-ends goals can be full boolean expressions with negations and disjunctions as well as conjunctions. While other planning methods can find the best solutions to planning problems, means-ends provides a more intuitive basis to planning and a good predictor of human behavior.

Means-ends planning easily handles unexpected events: Just replan with the original goal conditions. So stochastic effects of operators are naturally accommodated. Replanning can also be made efficient by caching recommended actions for subproblems as they are discovered [9], so caching of subplans is not necessary as with some other methods [12]. Stochastic effects can be a probability of failure of an action (as of trying to extinguish a fire), modification of state (as when a fire that is out flares up again), a mistake by an agent (as when someone mistakenly turns the power back on when the fire is not confirmed to be out), or just a random variable associated with the duration of each action. Such effects can also be made situationally dependent, so that the probability of an explosion is higher while the power is on. We have used random events in many tutors for procedural skills in military-training tasks, where indeterminancy teaches students how to respond to a wide variety of crises that would be costly to simulate without a computer.

We extend stochastic means-ends analysis to real-time multi-agent paradigms by associating each action with a priority list of agents qualified to accomplish it. Agents represent different people (as on a team) and physical processes (such as fire and flooding); animate agents change states by planning whereas inanimate agents change states by difference equations. Agents plan independently to achieve their goals, assuming cooperation as necessary from other agents. They have skill levels for each task which are parameters in the stochastic process of calculating a duration and success probability for an action. Agents have resource limitations in that they cannot do more than one operator in the same time interval, and certain resources (such as a fire hose) cannot be shared. Arbitrary computations may be embedded to define the state changes of inanimate agents.

Agents can have both preassigned responsibilities (the electrician is responsible for electrical devices) as well as dynamically assigned responsibilities (like holding the fire hose). An agent can be active (doing a task), idle (if its goals are achieved), or waiting (if its goals are not achieved but it has nothing to do). Since we are primarily interested in modeling task-related teams, dynamic assignment is done in our model by an order-report paradigm. A superior gives an order to a particular idle subordinate to accomplish a particular set of goal conditions. The order "wakes up" the

subordinate agent if it is idle. It then constructs its own plan to accomplish the goals, executes the plan, reports back to its superior, and the initiating order is deleted from the state. The order-report paradigm permits modeling of incomplete knowledge by agents. In some cases the subordinate agent may require the help of an assistant (as in trying to extinguish a fire using a bulky hose) which requires sub-orders to the assistant.

The simulation of actions also adjusts the states with results of concurrent actions that terminated during the time interval of the original action. It permits actions to be aborted by other actions in high-priority circumstances and when preconditions become false. For instance, if a fire reignites when a crewman is ventilating a compartment, the crewman must cease ventilation and initiate extinguishment.

MEAGENT is easy to adapt to new applications by providing definitions of new operators and agents through our tools [9]. They eliminate the need of experts to program to implement a simulation. A Prolog reasoning engine provides essential automatic backtracking during plan construction. MEAGENT is suitable for many team activities, such as office-task automation, manufacturing, routine military procedures, and sports teams.

2. An Example: Ship Firefighting

We describe an application of MEAGENT to firefighting on U.S. Navy ships, a critical skill with which all ship personnel must be familiar [11]. Since a physical experiment is risks human lives and expensive resources, agent-based simulation is essential to find bottlenecks and potential problems. (Critical-path analysis does not work well when there are significant stochastic effects or complex logical dependencies between actions, both characteristics of this and many other emergency-response tasks.) Generally the cost of a firefighting plan is assessed as the total time to complete it.

The agents in this simulation are the fire-team leader (or "scene leader"), the members of the fire team including at least an electrician, nozzleman, and hoseman, the command center monitoring alarms and giving orders to the team leader, and the fire itself. Figure 1 gives the simplest deterministic means-ends tree when no interruptions or surprises occur (although this is rare in both real firefighting and our tailored simulation) [6, 10].

Random events include casualties, availability of a medic, communication failures, equipment malfunctions, and whether the oxygen is safe when tested after the fire is out. Action durations are random variables depending on the skill level of the assigned agent, the fire size, the kind of tool used, smoke intensity, and water magnitude. For instance, the time to desmoke is a uniform random variable with mean 0.8*(Smoke/Skill) and standard deviation 0.75 of that.

The fire agent uses a stochastic epidemic model of fire growth that is applied to one-minute time steps. Its parameters are the fire type (there are four standard ones) and amount of flammable material, ignition and burnout rates, and the conditions affecting burnout (self-extinguishment) and flashover (sudden combustion of all flammable materials when the temperature gets very high). If the fire size exceeds a low threshold, an alarm sensor is signaled in the Command Center of the ship, and they order the fire team to the scene. Extinguishing is modeled by a negative ignition rate except when the wrong method is being used. Rates have random fluctuations to reflect the variety of different materials available to burn and the complexities of fire spread.

3. Plan Analysis for Counterplanning

Obstructive counterplanning is planning to interfere with or frustrate an existing plan [3]. There are two important kinds: Counterplanning in an adversarial situation, as in military defensive planning, and planning in an educational environment, as in the choice of obstacles presented by the system to the user in game-like tutors. In both cases the counterplan can be rated on the induced change to the cost of the plan, typically measured in time, physical effort, mental effort, or some combination thereof. In adversarial counterplanning the objective is to find the counterplan with largest benefit (negative effect on plan cost) minus counterplan cost, taking any uncertainty into account via decision theory. In educational counterplanning the objective is usually to provide accidents and unexpected conditions that require replanning that is neither too easy nor too hard, as well as manifesting a degree of variety. [2] endorses hierarchical planning much like means-ends as a good basis for planning in adversarial situations.

3.1 Analysis of Skills Criticality

One way to attack a multiagent plan is to interfere with the ability or readiness of agents to participate in it. But a good counterplanner should apportion their resources carefully since not all abilties are equally important. MEAGENT provides an tool to conduct experiments to guide such resource allocation.

For example, experiments assessed teams with different skill levels in firefighting [1]. 100 trials were done for 27 skill combinations (of high, medium, and low values) for a fire team of four members where the nozzlemen and the hosemen have the same skill levels. Two kinds of fire locations were used, a highly inflammable compartment and a more typical compartment. We measured performance of a team as the time elapsed between the first appearance of the fire and the recording of the completion of debriefing by the Command Center. An upper limit on simulation duration of 400 minutes was enforced for when the fire recurs numerous times. Figure 2 shows typical data, showing that the tail of the distribution is not Gaussian.

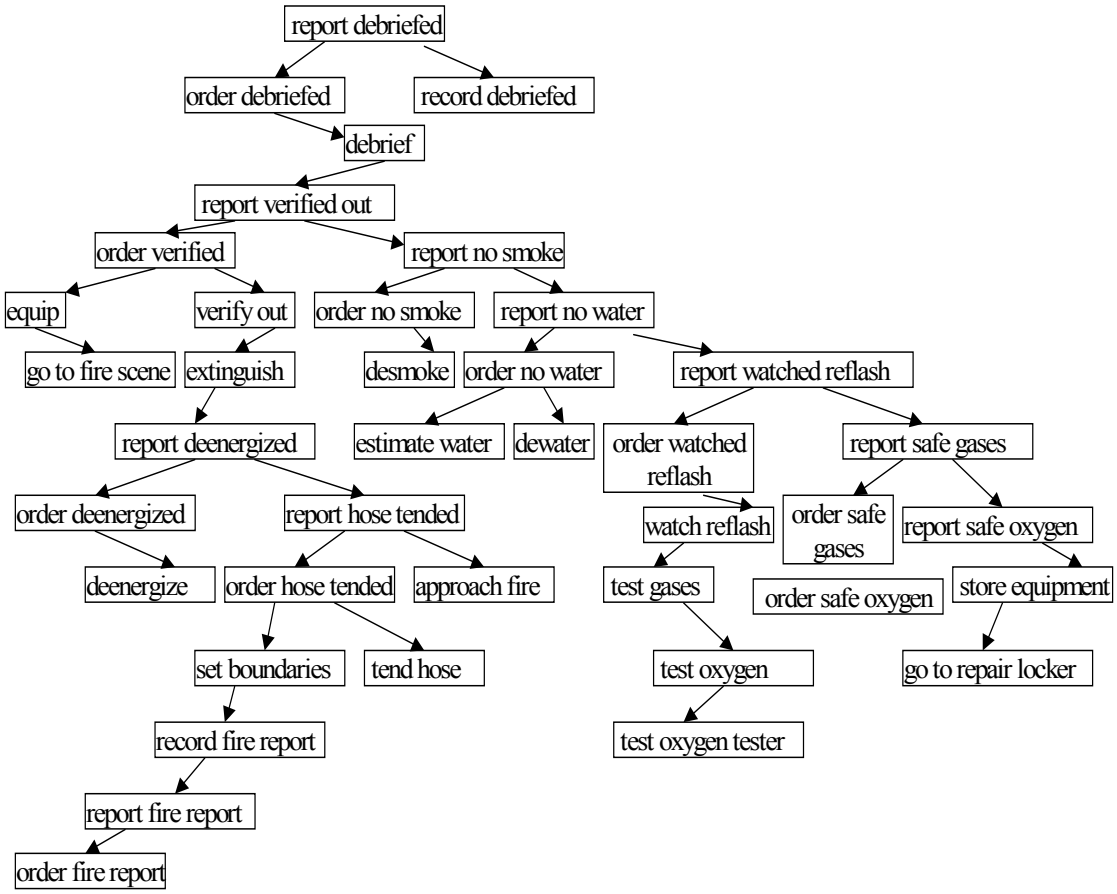

Figure 1: Plan tree for firefighting in a single compartment with no unexpected events.

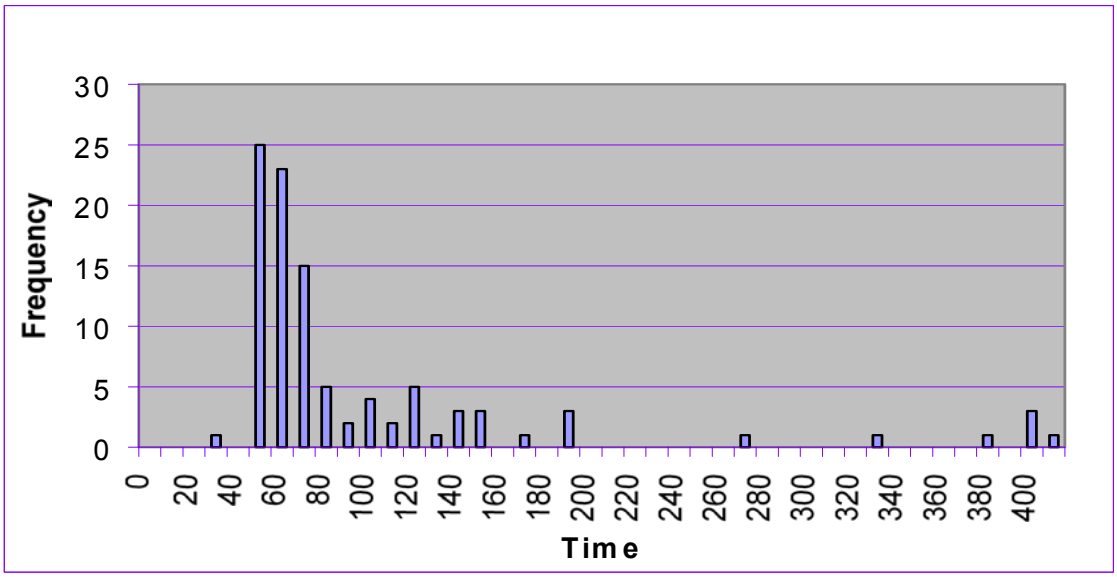

Figure 2: Histogram for total time for 100 runs of firefighting for ignition rate=0.5, burn-out rate=0.25, and all members with skill level 0.1.

Experiments permitted concrete guidelines for staffing policies for Navy fire teams:

- When the fire spreads faster, and the amount of material that burns out is very high, the percentage of intact inflammables at the end of action is going to be low, no matter what the composition of a team. Otherwise, performance is sensitive to team skill levels.
- The skills of the electrician did not much affect overall team performance; skills for the hosemen/nozzlemen were critical; and skills of the scene leader were important but not critical.

3.2 Systematic Counterplanning

But in general, counterplanning should try to change nonnumeric aspects of a plan, as fixing matters by the planner often then requires both additional thinking and time. MEAGENT provides a way to do such "systematic counterplanning": (1) Create a standard "base" plan; (2) change states within the plan by one fact each in every possible way (i.e., investigate counterplanning "ploys"), and replan to accomplish each agent's goals after each ploy; (3) infer the degree of damage due to each ploy; (4) construct counterplans to accomplish each ploy and calculate their costs to the counterplanner; and (5) select the best cumulative counterplan that accomplishes the best set of ploys at the minimum cost. Step (2) is analogous to the idea of partial derivatives of a continuous function. If the base plan involves S states, and the average state can be modified in M ways, we must replan MS times to assess the effects of each change. A replanning from state K from the end of the base plan involves selecting an action at each of K states, so systematic counterplanning requires about $0.5MS^2$ action selections for a single linear plan. This approach is much more suitable for machine implementation than the very-general top-down approach of [3], which is mainly intended for understanding narrative accounts of counterplanning, or the general criteria for organizational maladaptivity of [4], which propose subgoals like that of discouraging intra-organization communication without suggesting specific mechanisms to do so.

To find all possible ploys, pairs of opposite facts can be inferred if the opposite fact is always added whenever an operator or random change deletes the original fact, with the additional condition that the two facts never appear together in the same state or operator condition list. For example, "safe(gases)" is an opposite of "unsafe(gases)". Otherwise, facts that are deleted by at least one operator or random change are considered facts deletable from any state in which they occur, and facts that are added by at least one operator or random change are considered facts addable to any state. In firefighting, "raging(fire)" is deletable by extinguishment, and "smokey" is addable by the fire agent for a new fire. In addition, a good counterplanner will try if possible add facts that have not been considered in any of the planner's plans, such as fires in new locations or physical obstructions, but a good planner should be robust and have anticipated such changes, so we do not consider them here.

A problem in inferring possible changes to states is that good planning generally involves generalized reasoning with methods using variables. Generality can be preserved with deletions, but for additions and changes to states we must instantiate most variables since states must be concrete. We do this by inferring, for each variable that can appear in a state, its possible instantiations. For instance in firefighting, "ordered(F,P1,P2)" specifies that person P1 has ordered person P2 to make fact F become true; P1 and P2 must be instantiated to people in a superior-subordinate relationship, and F must be instantiated to a fact or its negation that can be accomplished. Using the firefighting specifications for instance for states for a single fire at a single location and its cleanup, we found after instantiation around 93 possible changes to a random state consisting of around 6 changes of facts to their opposites, 26 deletions of facts, and 61 additions of facts.

Since MEAGENT is intended for stochastic models, we must conduct many runs with the same starting state and goals to assess a change – at least 100 for firefighting, as discovered in the skills-effect analysis. We also need to try different base plans created by different random choices on which to make changes to states, since qualitatively different states are encountered in different plans -- in firefighting we found 100 runs were necessary on the average to see one flashover event. Since the average firefighting plan involves 41 animate-agent actions and 96 states (the fire and random occurrences creates state changes on their own), full counterplanning analysis of firefighting involves 0.5*93*96*96*100*100 = 4,285,440,000 action selections. At our typical 0.05 seconds per selection in Gnu Prolog on a two-year-old machine, this requires an impractical 2290 days, although we could sample the change space to approximate a solution. This is just for a single fire, and many more states are possible for multiple fires.

But if we proceed with this approach, after each set of runs with the same parameters we will have a distribution of costs associated with each kind of change. Cost distributions that are significantly different from one another are likely to reflect meaningful logical distinctions worth investigating. The technique of analysis of variance can identify such pairs if they are normally distributed. Otherwise, as for emergency-oriented planning like that shown in Figure 2, we may be able to partition the normal part of the distribution from a tail and analyze it separately. Doing that for firefighting by making changes to states at representative times in a representative run and then

replanning, we found 69% of the changes resulted in behavior more than two standard deviations away from normal. An equal number were cost-increasing and cost-decreasing, with the latter becoming more common towards the end of the plan.

3.3 Efficient Counterplanning

Systematic counterplanning can be made much more efficient by reasoning to eliminate redundant experiments. One idea we use is to collapse analysis of identical states in different runs, building a Markov state model with state-transition probabilities. State identity can be qualitative to further reduce the possibilities. In firefighting for instance, differences in fire, smoke, and water size do not affect planning. So we found only 2,496 distinct states in 60,360 actual states in 500 runs (and 2,166 in 250 runs), which shortens our work by a factor of 24.2.

A second efficiency idea is to logically infer sets of states that require identical responses to a counterplanning ploy. Typically, a single ploy affects only a few actions in a plan. Thus the ploy will not affect planning within a period of time after the state to which it is applied – this is what makes good planning still valuable in environments with significant uncertainty. Let ploy C be a single change made to a state, either a deletion of a fact D, an addition of a fact A, or change of D to its opposite A. Let the "fix plan" be the plan to respond to ploy C by directing the agents back to a known state. Then three inference methods apply. (1) *"Forward temporal inheritance": Action X in previously completed plan P is unaffected by prior ploy C at state S if X and all actions between S and X do not require D, do not require not (A), do not add fact D, or do not delete fact A. (2) "Forward temporal ploy cancellation": Actions in a previously completed plan P have a null fix plan for ploy C if they follow or are identical to an action X after C that adds fact A or deletes fact D. (3) "Plan following": States after ploy C resulting from performing one action in its fix plan have a new fix plan which is the rest of the actions in the original fix plan.* For example, consider the ploy, in firefighting after the fire is out, of a saboteur preventing the electrician from reporting to the scene leader that the fire area is desmoked. Such a report is only necessary as a precondition to having the fire team return to the repair locker, so the "fix plan" would be to wait until that point when the missing information is realized and have the scene leader find the electrician and ask about the smoke (or alternatively check the fire area themselves, but this would undercut the electrician's responsibilities and would be an inferior fix plan). This fix plan will temporally inherit forward from any state after the fire is out up to the action of returning to the repair locker, including all states resulting from random events. The fix plan will be cancelled if a random event occurs that the electrician manages to report the desmoking. A plan-following inference would occur if some possible action before returning to the repair locker results in finding the electrician, subsequent states of which can then infer a simpler fix plan of just asking the electrician if the fire area is desmoked.

Our inference rules apply to several kinds of fix-planning strategies. The two most obvious are to immediately plan to undo the effects of C before resuming the original plan (a "lazy repair") or to plan to achieve the original goals from the new state ("radical"). Repair fix-plans will be significantly faster to build than radical fix-plans in most cases, but may result in suboptimal overall behavior because ploy C may affect reasons for the actions in the original plan, while not affecting its adequacy. Adequacy usually suffices because people are creatures of habit and prefer to persist with now-suboptimal original plans. However, if C results in the final state of the plan not satisfying the original goals after repair, more replanning must be done even with lazy repair.

The inference methods suggest a ploy taxonomy. Ploy C can be ignored after a state if temporal inheritance of an empty fix plan extends to the end of the plan. For instance in firefighting, deletion of the fact that the oxygen tester itself has been tested makes no difference after the fire is out and the oxygen is found to be safe. Ploy C can also be ignored if it reduces the cost of the plan. For instance, turning off the power at the fire scene is doing something that always needs to be done and does not hurt doing early. Depending on the application, ploy C may also be ignored if it is undoable by a single action (i.e., if the lazy repair plan is trivial). For instance, the ploy of turning the power on can be easily undone by turning it back off. All ploys not in these three categories can be thus considered nontrivial to address.

For our example of firefighting, we found 93 possible ploys, which interfered with 4.3% of the states in an average plan; 2.8% of these involved violation of conditions associated with an action applied to that state, and 1.5% were cancelled by an action. Our three kinds of inferences found an average of 1.2 additional states for which a fix plan could be used; in addition, fix plans need only be calculated to reach known states, reducing the fix-plan planning effort to 36% of the typical original planning effort at a state. This reduction in effort of 16.2% combined with the reduction due to the Markov state model reduced the effort of counterplanning to 0.67% of that of the systematic approach, reducing expected time from 2290 to a 15.3 days, making it practicable.

3.4 Using Costs in Counterplanning

Finally, after promising changes to a plan have been determined and their costs computed, we can formulate a counterplan to most cost-effectively foil the plan. We cannot generalize too much here because counterplanning requires predominantly different operators than planning (including deception), as suggested by the heuristics in [3] and illustrated in attacks on computer systems [10]. For instance, a firefighting saboteur can turn the power off, make

equipment inoperable, pour gasoline around, disconnect the hose during extinguishment and plead ignorance, report incorrectly that they have checked oxygen, or relay incorrect or fake orders to subordinates. But such actions are only possible surreptitiously, thus requiring costly waiting time until circumstances are favorable and a particular probability of failure to account for. There are also potential nonlinear effects for the counterplanner to deal with, such as that one successful sabotage may make the next one harder as agents become aware that something is wrong, or that minor independent changes to a state may cause an agent to question their memory and become overly cautious. These nonlinear effects can be very significant; so since communications in crises are always problematic, communications ploys like ignored orders may be among the most successful deceptions.

However the tactics are selected, we can evaluate the counterplan costs and compare them to the benefits of sets of damaging changes to the plan. A* search is a classic way to do this tradeoff in combinatorial problems, and game theory can be used to plan for counter-responses to counterplans, and so on. Besides firefighting, we are currently investigating models of aircraft-carrier flight-deck operations and attacks by hackers on computer systems.

References

[1] S. Andrade, N. Rowe, D. Gaver, and P. Jacobs, Analysis of shipboard firefighting-team efficiency using intelligent-agent simulation, *Proc. Command and Control Research and Technology Symposium*, Monterey CA USA, June 2002.

[2] C. Applegate, C. Elsaesser, and J. Sanborn, An architecture for adversarial planning, *IEEE Transactions on Systems, Man, and Cybernetics*, 20 (1), January/February 1990, 186-194.

[3] J. Carbonell, Counterplanning: A strategy-based model of adversary planning in real-world situations, Artificial *Intelligence*, 16, 1981, 295-329.

[4] K. Carley, Inhibiting adaptation, *Proc. Command and Control Research and Technology Symposium*, Monterey CA USA, June 2002.

[5] K. Carley, J. Kjaer-Hansen, A. Newell, and A. Prietula, Plural-Soar: A prolegomenon to artificial agents and organizational behavior, in M. Masuch and G. Massimo (Eds.), *Distributed Intelligence: Applications in Human Organizations* (Amsterdam: North-Holland, 1992), 87-118.

[6] Federal Emergency Management Agency, United States Fire Administration, *Developing Effective Standard Operating Procedures for Fire & EMS Departments*, no date.

[7] A. Law and D. Kelton, *Simulation Modeling and Analysis, Second Edition* (New York: McGraw-Hill, 1991).

[8] A. Newell and H. Simon, GPS, A program that simulates human thought, in E. Feigenbaum and J. Feldman (Eds.), *Computers and Thought* (Berlin: R. Oldenburg KG, 1963).

[9] N. Rowe and T. Galvin, An authoring system for intelligent procedural-skill tutors," *IEEE Intelligent Systems*, 13(3), May/June 1998, 61-69.

[10] N. Rowe, and S. Schiavo, An intelligent tutor for intrusion detection on computer systems, *Computers and Education*, 31, 1998, 395-404.

[11] U.S. Navy, *Naval Ships' Technical Manual Chapter 555 – VOLUME 1-Surface Ship Firefighting*, sections 1-9, 1-13, 1-22, 1-23, 1-24, and 9-1, 1998.

[12] G. Weiss (Ed.), *Multiagent Systems* (Cambridge, MA: MIT Press, 1999).